# Climate Specific Tools: The cdutil Package

# cdutil - overview

- The cdutil Package contains a collection of sub-packages useful to deal with Climate Data
- Sub-components are:
  - **times**: a collection of tools to deal with the time dimension.
  - **region**: a “region” selector for rectilinear grids
  - **vertical**: already seen earlier in the course
  - **averager**: already dealt with earlier in the course
  - **continents_fill**: Emulate a VCS graphic method to display filled continents, see docs.
  - **VariableConditioner** and **VariablesMatcher**: A superset of the regridder and time extraction tools, see CDAT docs.

# cdutil - “times” module (1)

- **cdutil.times** for time axes, geared toward climate data.

- All **seasonal** extractions in this module are based on “**bounds**”, **times** provides functions to set the bounds correctly (remember: time axis doesn’t have any bounds unless they are in the file).

- These functions are:

```
cdutil.times.setTimeBoundsMonthly(slab/axis)
cdutil.times.setTimeBoundsYearly(slab/axis)
cdutil.times.setTimeBoundsDaily(slab/axis,frequency=1)
```

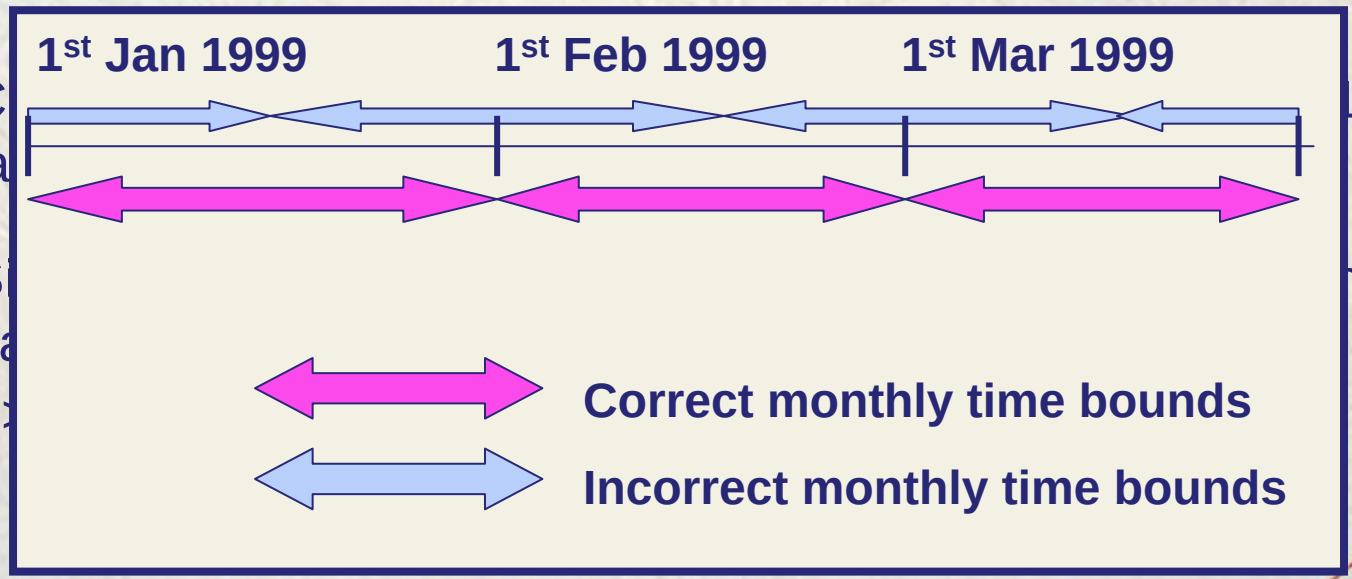
- ***Important note:*** **cdutil** imports everything in the **times** module so you can just call e.g.:

```
cdutil.setTimeBoundsMonthly(slab/axis)
```

# The importance of understanding bounds

- CDAT used to set bounds automatically. E.g.:
  `longitude = [0, 90, 180, 270]`

  ∴ `bounds = [[-45, 45], [45, 135],`
  `[135, 225], [225, 315]]`
- Seems reasonable, but imagine a monthly mean time series where the times are recorded on 1st day of each month:
  `timeax=["1999-1-1", "1999-2-1", …, "2100-12-1"]`
- C… …15th
  la…
- Si… …he
  da…
  >>…



# Temporal averaging

- Averaging over time is a special problem in climate data analysis.
- **cdutil** makes the extraction of time averages and climatologies simple.
- Functions for annual, seasonal and monthly averages and climatologies
- User-defined seasons (such as “FMA” = Feb/Mar/Apr).

# Pre-defined time-related means

- DJF, MAM, JJA, SON (seasons)

```
>>> djf_mean=cdutil.DJF(my_var)
```

- SEASONALCYCLE (means for the 4 predefined seasons [DJF, MAM, JJA, SON ]) – array of above.

```
>>> seas_mns=cdutil.SEASONALCYCLE(my_var)
```

- YEAR (annual means)
- ANNUALCYCLE (monthly means for each month of the year)
- Additional arguments can be passed, the default needs 50% of the season to be present order to assign a value.

# Climatologies and departures (1)

Season extractors have 2 functions available:

- **climatology**: which computes the average of all seasons passed. `ANNUALCYCLE.climatology()`, will return the 12 month annual cycle for the slab:

```
>>> ann=cdutil.ANNUALCYCLE.climatology(v)
```

- **departures**: which given an optional climatology will compute seasonal departures from it.

```
>>> d=cdutil.ANNUALCYCLE.departures(v, cli60_99)
```

# Note that the second argument is optional but can be a pre-computed climatology such as here *cli60_99* is a 1960-1999 climatology but the variable *v* is defined from 1900-2000. If not given then the overall climatology for *v* is used.

# Climatologies and departures (2)

To calculate long-term averages (over multiple years):

- DJF.climatology(), MAM.climatology() etc.,

```
>>> djf_clim=cdutil.DJF.climatology(my_var)
```

- SEASONALCYCLE.climatology() - climatologies for the 4 predefined seasons [DJF, MAM, JJA, SON ] – array of above.
- YEAR.climatology() - annual mean climatologies.
- ANNUALCYCLE.climatology() - 12 climatologies, one per month
- ***Note:*** **You can replace any of the above to calculate the departure from the climatology**

# cdutil - “region” module (1)

- The **cdutil.region** module allows the user to extract a region “exactly”. i.e. resetting the latitude and longitude bounds to match the area “exactly”, therefore computing an “exact” average when passed to the averager function.

- Predefined regions are:
  - AntarcticZone, AAZ (South of latitude 66.6S)
  - ArcticZone, AZ (North of latitude 66.6N)
  - NorthernHemisphere, NH # useful for dataset with latitude crossing the equator
  - SouthernHemisphere, SH
  - Tropics (latitudes band: 23.4S, 23.4N)

# cdutil – “region” module (2)

- Creating your selector:

```
myselector=cdutil.region.domain(latitude=(lat1,lat2), longitude=(lon1,lon2)) # can be any dimension, but very useful for lat/lon
```

- Using the selector:

```
slab2=slab1(myselector)
```